Unix/Linux Command Reference

FOSSwire.com

File Commands

ls – directory listing

ls -al – formatted listing with hidden files

cd *dir* - change directory to *dir*

cd – change to home

pwd – show current directory

mkdir *dir* – create a directory *dir*

rm *file* – delete *file*

rm -r *dir* – delete directory *dir*

rm -f *file* – force remove *file*

rm -rf *dir* – force remove directory *dir* *

cp *file1 file2* – copy *file1* to *file2*

cp -r *dir1 dir2* – copy *dir1* to *dir2*; create *dir2* if it doesn't exist

mv *file1 file2* – rename or move *file1* to *file2* if *file2* is an existing directory, moves *file1* into directory *file2*

ln -s *file link* – create symbolic link *link* to *file*

touch *file* – create or update *file*

cat > *file* – places standard input into *file*

more *file* – output the contents of *file*

head *file* – output the first 10 lines of *file*

tail *file* – output the last 10 lines of *file*

tail -f *file* – output the contents of *file* as it grows, starting with the last 10 lines

Process Management

ps – display your currently active processes

top – display all running processes

kill *pid* – kill process id *pid*

killall *proc* – kill all processes named *proc* *

bg – lists stopped or background jobs; resume a stopped job in the background

fg – brings the most recent job to foreground

fg *n* – brings job *n* to the foreground

File Permissions

chmod *octal file* – change the permissions of *file* to *octal*, which can be found separately for user, group, and world by adding:

- 4 – read (r)
- 2 – write (w)
- 1 – execute (x)

Examples:

chmod 777 – read, write, execute for all

chmod 755 – rwx for owner, rx for group and world

For more options, see **man chmod**.

SSH

ssh *user@host* – connect to *host* as *user*

ssh -p *port user@host* – connect to *host* on port *port* as *user*

ssh-copy-id *user@host* – add your key to *host* for *user* to enable a keyed or passwordless login

Searching

grep *pattern files* – search for *pattern* in *files*

grep -r *pattern dir* – search recursively for *pattern* in *dir*

command* | grep *pattern – search for *pattern* in the output of *command*

locate *file* – find all instances of *file*

System Info

date – show the current date and time

cal – show this month's calendar

uptime – show current uptime

w – display who is online

whoami – who you are logged in as

finger *user* – display information about *user*

uname -a – show kernel information

cat /proc/cpuinfo – cpu information

cat /proc/meminfo – memory information

man *command* – show the manual for *command*

df – show disk usage

du – show directory space usage

free – show memory and swap usage

whereis *app* – show possible locations of *app*

which *app* – show which *app* will be run by default

Compression

tar cf *file.tar files* – create a tar named *file.tar* containing *files*

tar xf *file.tar* – extract the files from *file.tar*

tar czf *file.tar.gz files* – create a tar with Gzip compression

tar xzf *file.tar.gz* – extract a tar using Gzip

tar cjf *file.tar.bz2* – create a tar with Bzip2 compression

tar xjf *file.tar.bz2* – extract a tar using Bzip2

gzip *file* – compresses *file* and renames it to *file.gz*

gzip -d *file.gz* – decompresses *file.gz* back to *file*

Network

ping *host* – ping *host* and output results

whois *domain* – get whois information for *domain*

dig *domain* – get DNS information for *domain*

dig -x *host* – reverse lookup *host*

wget *file* – download *file*

wget -c *file* – continue a stopped download

Installation

Install from source:

```
./configure
make
make install
```

dpkg -i *pkg.deb* – install a package (Debian)

rpm -Uvh *pkg.rpm* – install a package (RPM)

Shortcuts

Ctrl+C – halts the current command

Ctrl+Z – stops the current command, resume with **fg** in the foreground or **bg** in the background

Ctrl+D – log out of current session, similar to **exit**

Ctrl+W – erases one word in the current line

Ctrl+U – erases the whole line

Ctrl+R – type to bring up a recent command

!! - repeats the last command

exit – log out of current session

* use with extreme caution.

OpenFOAM Quick-Reference

.mavt.ethz.ch

Important Environment Variables

`$WM_PROJECT_DIR` – path to the OpenFOAM installation

`$WM_PROJECT_USER_DIR` – OpenFOAM user directory

`$FOAM_TUTORIALS` – OpenFOAM tutorials

`$FOAM_SRC` – source-tree of OpenFOAM libraries

`$FOAM_APP` – source-tree of OpenFOAM applications

`$FOAM_APPBIN` – directory with the applications

`$FOAM_USER_APPBIN` – directory with the applications created by the user

`$FOAM_LIBBIN` – directory with the libraries provided by OpenFOAM

`$FOAM_USER_LIBBIN` – directory with the libraries created by the user

`$FOAM_RUN` – directory where the user can put his/her cases

Important Shell-Aliases

`run` – *cd* to `$FOAM_RUN`

`src` – *cd* to `$FOAM_SRC`

`app` – *cd* to `$FOAM_APP`

`util` – *cd* to `$FOAM_APP/utilities`

`sol` – *cd* to `$FOAM_APP/solvers`

Definitions Used Here

`case` – relative or absolute path to the case

Basic Case Structure

```
case/ – the case directory
+ 0/ – contains initial and boundary conditions
+ constant/ – constant data
  + polyMesh/ – contains the grid data
  + transportProperties – viscosity
+ system/ – run-time control / numerics
  + controlDict – run-time control
  + fvSchemes – numerical schemes
  + fvSolution – linear solvers
```

`case`**`/0/`** – contains for each variable a file defining the initial and boundary conditions. May also contain initial and boundary conditions for a moving grid.

`case`**`/constant/polyMesh/`** – contains the grid data for a non-moving grid. The files are: `boundary`, `faces`, `neighbour`, `owner`, `points`.

`case`**`/constant/transportProperties`** – defines the viscosity (also for non-Newtonian fluids)

`case`**`/system/controlDict`** – sets start-/endtime, time-step size, output control etc. Also allows to load general “plugins” and apply “function-objects” to compute forces acting on a surface.

`case`**`/system/fvSchemes`** – defines the numerical schemes to be used for each differential operator

`case`**`/system/fvSolution`** – selects the solvers to be used for the linear equation systems for each variable which is solved for using an implicit scheme.

Uniform Invocation Syntax

`app` **`-case`** ***`case`*** – start *app* (the case is in *case*). Sometimes more options/arguments are accepted.

`app` – start *app* directly if you are already in *case*

`app` ***`-help`*** – display short help message for *app*

`app` ***`-doc`*** – open documentation for *app*

`app` ***`-srcDoc`*** – open source-documentation for *app*

Important (Basic) Solvers

`laplacianFoam` – solve Laplace equation, suitable for e.g. thermal diffusion in solids

`potentialFoam` – potential flow solver, suitable for generating good initial conditions

`scalarTransportFoam` – solve scalar transport equation, suitable for e.g. postprocessing

`icoFoam` – solves the incompressible Navier-Stokes equations for Newtonian fluids (for laminar flows)

`turbFoam` – solves the incompressible RANS equations using turbulence modelling for Newtonian fluids (turbulent flows)

`icoDyMFoam/turbDyMFoam` – for dynamic mesh cases

`simpleFoam` – steady-state solver for the incompressible Navier-Stokes equations for non-Newtonian fluids

`interFoam` – solver for 2 incompressible, immiscible fluids

Important Utilities

`blockMesh` – creates the mesh defined by *case/constant/polyMesh/blockMeshDict*

`decomposePar` – splits the case for parallel run, controlled by *case/system/decomposeParDict*

`reconstructPar` – reassembles the decomposed solution of a parallel run

`paraFoam` – starts ParaView to visualize the results

`touch` ***`case/caseName`*****`.foam && paraview --data=`*****`case/caseName`*****`.foam`** – starts ParaView with an alternative (better) plugin

Structure of a Solver

```
appName/ – the directory with the source code
+ appName.C – the main program
+ createFields.H – declarations and initializations of all fields
+ Make/ – compilation instructions
  + files – list of source/output files
  + options – compilation options
```

`appName`**`/`*****`appName`*****`.C`** – the actual solver code

`appName`**`/createFields.H`** – declares all the field variables and initializes the by (usually) reading the initial conditions from a file.

`appName`**`/Make/files`** – names all the source (.C) files, one file per line. The last line should read `EXE=$(FOAM_USER_APPBIN)/`*appName* to specify the name and location of the output file.

`appName`**`/Make/options`** – specifies directories to search for include files and libraries to link the solver against. The former are specified in the variable EXE_INCS, the latter in EXE_LIBS. Lines have to be continued using the \ character